High Level design:

# Network Protocol:

1. Application Layer & Transport Layer

Client Server Protocol:

* HTTP
* FTP
* SMTP
* Web Socket

Peer to Peer:

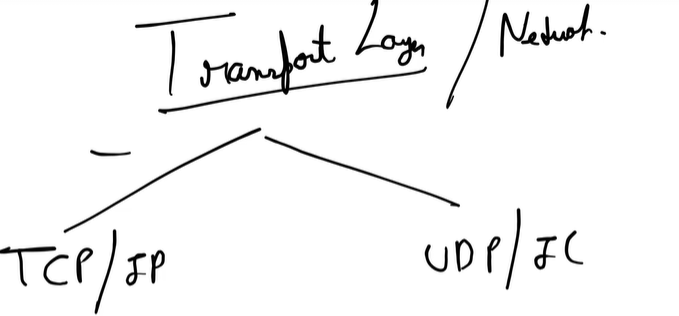
* WebRtc

Peer to peer : Where 2 clients will communicate with each other that is known as peer to peer

Client server : Where clients communication happens with the Server itself.

WebSocket is usefull when we need to desing the Chatting applications.

## Transport Layer:



* In TCP/IP we will create the virtual connections
* 3 way handshaking and ack was provided in TCP Connections

UPD:   
will not maintain the virtual connection

* Ordering is not maintain in UDP
* Fast in processing
* No ack we will provide in UDP
* WEBRTC Protocol uses the UDP for peer to peer connection

## CAP Theorem:

Consistency, Availability, Partition Tolerance

Desirable property of distributed System

## Microservice Design Pattern 1:

Monolithic also known as legacy service:

* Overloaded disadvantage as we have everything with one service only
* Overload IDE
* Scaling is hard and we can not do the changes easily
* For minimum changes we need to deploy the whole application
* If we have increasing load on order services then we have to replicate the whole application to other services in monolithic.
* All the disadvantage is a advantage of Micro services

Disadvantage of MS:

* Properly division of services must be happen and functionality should be define properly
* All the services are loosely coupled.
* Latency can increase if there is too much dependency in between the classes
* If we do the changes in one of the services then it can break the other service as well.
* **Transaction Management:** If we have a single db then we can follow the ACID   
  But in MS Each services have there own DB, and if one request need to follow the call of other services transaction then we need to handle that in other services as well.

**Phases of Microservices:**

1. **Decomposition**
2. **Data Driven**
3. **Communication**
4. **Integration**